

# Refining Approximating Betweenness Centrality Based on Samplings

Shiyu Ji, Zenghui Yan  
 {shiyu,zyan}@cs.ucsb.edu

## Abstract

Betweenness Centrality (BC) is an important measure used widely in complex network analysis, such as social network, web page search, etc. Computing the exact BC values is highly time consuming. Currently the fastest exact BC determining algorithm is given by Brandes, taking  $O(nm)$  time for unweighted graphs and  $O(nm + n^2 \log n)$  time for weighted graphs, where  $n$  is the number of vertices and  $m$  is the number of edges in the graph. Due to the extreme difficulty of reducing the time complexity of exact BC determining problem, many researchers have considered the possibility of any satisfactory BC approximation algorithms, especially those based on samplings. Bader et al. give the currently best BC approximation algorithm, with a high probability to successfully estimate the BC of one vertex within a factor of  $1/\varepsilon$  using  $\varepsilon t$  samples, where  $t$  is the ratio between  $n^2$  and the BC value of the vertex. However, some of the algorithmic parameters in Bader's work are not yet tightly bounded, leaving some space to improve their algorithm. In this project, we revisit Bader's algorithm, give more tight bounds on the estimations, and improve the performance of the approximation algorithm, i.e., fewer samples and lower factors. On the other hand, Riondato et al proposes a *non-adaptive* BC approximation method with samplings on shortest paths. We investigate the possibility to give an *adaptive* algorithm with samplings on pairs of vertices. With rigorous reasoning, we show that this algorithm can also give bounded approximation factors and number of samplings. To evaluate our results, we also conduct extensive experiments using real-world datasets, and find that both our algorithms can achieve acceptable performance. We verify that our work can improve the current BC approximation algorithm with better parameters, i.e., higher successful probabilities, lower factors and fewer samples.

## 1 Introduction

In the past a few decades, how to evaluate the importance of each vertex in a network has attracted extensive researchers [6]. It is still a popular open problem that how to efficiently find the vertex who plays a central role given the network structure. Many measures of centrality have been proposed: Closeness Centrality [25], Graph Centrality [15], Stress Centrality [26], Betweenness Centrality [13]. Recently Betweenness Centrality (BC) has been popularly researched [4, 8, 1], and has many theoretical applications, e.g., fully dynamic graphs [17], all pairs shortest paths (APSP) and diameter problems [1], clustering [12], etc., as well as many real-world applications, e.g., delay-tolerant networks [20], deltaic river network [11], hot spots on crime data networks [27], etc.

Betweenness is a measure of centrality based on single source shortest paths (SSSP). We first give its definition.

**Definition 1.** Betweenness Centrality [13, 7, 4] of a vertex  $v$  in the graph  $G = (V, E)$  is

$$BC(v) = \sum_{s,t \in V, s \neq t \neq v} \delta_{st}(v), \quad (1)$$

where  $\delta_{st}(v)$  is called **pair-dependency** of vertex  $v$  given a pair  $(s, t)$ , and is defined as

$$\delta_{st}(v) = \frac{\sigma_{st}(v)}{\sigma_{st}}, \quad (2)$$

in which  $\sigma_{st}(v)$  denotes the number of the shortest paths from  $s$  to  $t$  that pass through  $v$ , and  $\sigma_{st}$  denotes the total number of the shortest paths from  $s$  to  $t$  ( $v$  does not have to be on the paths).

By the definition it is clear that the BC value must be between 0 and  $n^2$ .

We summarize our major contributions in this paper:

- We revisit Bader’s work and give our new sampling algorithm based on a mixed idea between Bader’s and Riondato’s works. Analysis and evaluation results are given.
- We find the bounds in Bader’s paper [4] are sub-optimal and some lemmas have gaps, e.g., 1) the proof of Lemma 1 is not given, and 2) in the proof of Lemma 3, the following incorrect argument was used:

$$\Pr\left[\left(X_1 - \frac{A}{n}\right) + \cdots + \left(X_k - \frac{A}{n}\right) \geq cn - \epsilon n\right] \leq \sum_i \Pr\left[X_i - \frac{A}{n} \geq cn - \epsilon n\right].$$

Clearly the sum of  $n$  random variables exceeds a value  $S$  does not imply one of these random variable exceeds this value  $S$ .

- To repair the theory based on our findings above, we develop a new probability theoretic model that can well explain the results in Bader’s paper. Our results not only agrees with the existing research works mostly, but also gives more refined bounds and better performance, e.g., lower factors and higher successful probabilities.

We organize this paper as follows. In the related works, we will discuss the most important existing techniques of finding the exact or estimated BC values given a large graph. Then we will revisit Bader’s algorithm and give our new algorithm sampling on pairs of vertices. For both algorithms, we will rigorously show the results that can achieve improved bounds on number of samplings and approximation factors. To evaluate the performance, we will discuss our experimental results in Section 7. Finally, future works and conclusions will be presented

## 2 Related Works

Brandes [7, 8] gives the state-of-the-art fastest algorithm to calculate the exact value of BC for each vertex in the graph. Given a graph  $G = (V, E)$ , Brandes’ algorithm takes at least  $O(nm)$  time for unweighted graphs and  $O(nm + n^2 \log n)$  time for weighted graphs, where  $n$  is the number of vertices  $|V|$  and  $m$  is the number of edges  $|E|$ . However, for very large and complex networks, the performance of Brandes’ algorithm may be unacceptable [4]. For example, in large social networks, Brandes’ algorithm may take considerably large amount of time, but we want to find the hot spots or trending timely [16]. There are some heuristic variants on Brandes’ algorithm, but the complexity has not seen significantly improvement [21].

To address the performance problem, people have turned to paralleled algorithms [19, 5, 22, 23], randomized approximations [4, 14, 22, 23, 9, 2, 3], etc. It turns out that sampling plays an important role in these algorithms. There are two categories on the state-of-art sampling approximation algorithms: uniform sampling (e.g., to sample nodes with identical probability) [4, 14, 24] and non-uniform sampling (e.g., each node has a different probability to be sampled) [22, 23, 9, 2, 3]. However, none of the existing methods can show dominating performance over the rest. Hence how to choose the samplings to achieve better performance is still an open problem [4]. Moreover, many BC approximations by samplings lack theoretical performance bounds, and thus an improved bound may also speed up the algorithm [14].

In this project, we will revisit Bader’s algorithm, which takes  $\epsilon t$  samples to estimate the BC values within a factor of  $1/\epsilon$ , where  $t$  is the ratio between  $n^2$  and the BC value of the vertex. We will discuss the performance bound of the algorithm, and refine the sampling parameters to achieve better results, e.g., fewer samples and lower factors. On the other hand, we will also use the idea of Riondato’s sampling on shortest paths [22]. We will propose and analyze a new adaptive BC approximation algorithm. We will also use simulations with real-world data to verify our results.

### 3 Brandes' Exact Computation of Betweenness Centrality

In this section we will talk about the main idea of Brandes' algorithm [7], which gives exact computation on betweenness centrality (BC) value of each vertex in the network. This is the fastest algorithm on exact BC computation currently.

Dependency of a source vertex on a given node is frequently used in our algorithms and discussions. Here we give the definition: [7]

**Definition 2.** Given a graph  $G = (V, E)$ , the **dependency** of a source vertex  $s \in V$  on a vertex  $v \in V$  is

$$\delta_{s.}(v) = \sum_{t \in V, s \neq t \neq v} \delta_{st}(v). \quad (3)$$

Based on Definition 1, the BC value of vertex  $v$  is

$$BC(v) = \sum_{s \neq v} \delta_{s.}. \quad (4)$$

Brandes finds a recursive way to calculate the BC value of  $v$ , by introducing *predecessors* of  $v$ .

**Definition 3.** Given a graph  $(V, E)$ , the **predecessors** of a vertex  $v \in V$  on a shortest path from  $s$  to  $v$  is a subset  $P_s(v) \subseteq V$  s.t.

$$t \in P_s(v) \Rightarrow (d(s, v) = d(s, t) + d(t, v) \wedge (t, v) \in E),$$

where  $d(s, t)$  denotes the length of a shortest path from  $s$  to  $t$ .

The following theorem motivates Brandes' algorithm. Its proof is in [7].

**Theorem 1.** Given a graph  $(V, E)$ , for any  $s, v \in V$ , we have

$$\delta_{s.}(v) = \sum_{t \in V, s.t. v \in P_s(t)} \frac{\sigma_{sv}}{\sigma_{st}} (1 + \delta_{s.}(t)). \quad (5)$$

The basic idea of Brandes' algorithm can be summerized as follows:

1. For each vertex  $s \in V$ , we calculate the shortest paths from  $s$  to other vertices, i.e., using Dijkstra's algorithm, which takes at least  $O(|E| + |V| \log |V|)$  time.
2. For each vertex  $s \in V$ , traverse the vertices in descending order of their distances from  $s$ , and accumulate the dependencies by Theorem 1. Each traversal takes  $O(|E|)$  time.

The total complexity  $T(n, m)$  of Brandes' algorithm, where  $n = |V|$  and  $m = |E|$ , is therefore

$$T(n) = O(n(m + n \log n) + nm) = O(nm + n^2 \log n).$$

Even though Brandes' is the fastest algorithm on finding exact BCs, its time complexity can be unacceptable in many practical scenarios, e.g., very big graphs or large scaled datasets. This motivates the research in BC approximation based on samplings.

### 4 Approximation on Betweenness Centrality Based on Adaptive Vertex Samplings

In this section, we give Bader's BC approximation by adaptive-sampling [4]. The basic idea of Bader's algorithm is that if we are only interested in a loose estimation of BC value, then we do not have to compute the shortest paths from each vertex  $s \in V$ . Instead we can take samples  $S \subseteq V$  ( $|S| \ll |V|$ ) and compute

the shortest paths on each sample  $s \in S$ . Then we can accumulate the dependencies like how Brandes' does, and it is guaranteed that the final sum of dependencies is a good estimation of the BC value, i.e., the approximating factor can be provably bounded.

Consider we want to estimate the BC value of a vertex  $v$  given the graph  $G = (V, E)$ . We summerize Bader's algorithm as follows:

1. Let  $S = 0$ .
2. Sample a vertex  $v_i \in V$ . Compute the shortest paths from  $v_i$ . Accumulate the dependencies  $S \leftarrow S + \delta_{v_i.}(v)$ .
3. If  $S \leq cn$  for some  $c \geq 1$ <sup>1</sup>, repeat Step 2. Otherwise, output  $nS/k$  as the estimation of  $BC(v)$ , where  $k$  is the number of samples.

Bader shows this algorithm can successfully estimate the BC value with a high probability and its factor has an upper bound. The following theorem is the main result.

**Theorem 2. (Bader's)** *For any  $\varepsilon \in (0, 1/2)$ , Bader's algorithm can estimate  $BC(v)$  within a factor of  $1/\varepsilon$  with a successful probability no less than  $1 - 2\varepsilon$  and  $\varepsilon t$  samples.*

Bader's proof uses some loosely bounded arguments, e.g., the upper bound of the variance of the dependency  $\delta_{v_i.}(v)$  for each sample  $v_i$ . In this paper we will discuss the results when the arguments are more tightly bounded. In fact, we will show that the results in Theorem 2 can be improved.

## 5 Refining Results of BC Approximation Samplings

In this section we will discuss the probabilistic sampling model, and show an improved theorem which achieves better parameters compared to Theorem 2.

We first give the model assumptions on the samplings. Let  $X_i$  be the random variable of dependency  $\delta_{v_i.}(v)$ , i.e., the sampled source vertex is  $v_i$ , and the target vertex is  $v$ . Let  $A = BC(v)$ . The following lemma gives the probability distribution of  $X_i$ .

**Lemma 1.** *The cdf of  $X_i$  is*

$$\Pr[X_i \leq x] = \begin{cases} 0 & x \leq 0, \\ 1 - \left(1 - \frac{x}{A}\right)^{n-1} & 0 < x < A, \\ 1 & x \geq A. \end{cases}$$

*Therefore, the pdf of  $X_i$  is*

$$p(x) = \begin{cases} \frac{n-1}{A} \left(1 - \frac{x}{A}\right)^{n-2} & 0 < x < A, \\ 0 & \text{otherwise.} \end{cases}$$

*Proof.* See the Appendix. □

In the proof of Lemma 1, we assume the cutting points of the sub-intervals representing each  $X_i$  are uniformly distributed. This is the major assumption of our model. We will show that our model is highly compatible to the existing results, e.g., [4]

Now we can compute the mean  $E[X_i]$  and the variance  $Var[X_i]$ . Note a similar lemma has been proposed by Bader's paper (Lemma 2 in [4]).

**Corollary 1.**

$$E[X_i] = \frac{A}{n}, \quad Var[X_i] = \frac{n-1}{n^2(n+1)} A^2.$$

---

<sup>1</sup>Bader's paper [4] requires  $c \geq 2$ . We will see the lower bound of  $c$  can be relaxed to 1.

*Proof.* See the Appendix. □

Note that in [4], Lemma 2 states

$$E[X_i] = A/n, E[X_i^2] \leq A, \text{Var}[X_i] \leq A.$$

Since  $A \leq n^2$ , it is clear our bounds satisfy the cases of  $E[X_i]$  and  $\text{Var}[X_i]$ . We leave the discussion on the case of  $E[X_i^2]$  to the Appendix.

Like Bader's paper [4], we need two lemmas to show our main result.

**Lemma 2.** *Let  $k = \varepsilon(\frac{n^2}{A})^{\frac{2}{3}}$ . Then*

$$\Pr[\sum_{i=1}^k X_i \geq cn] \leq \frac{\varepsilon^3}{(c - \varepsilon)^2}.$$

*Proof.* See the Appendix. □

**Lemma 3.** *Let  $k \geq \varepsilon(\frac{n^2}{A})^{\frac{2}{3}}$  and  $d > 0$ . Then*

$$\Pr\left[\left|\frac{n}{k} \sum_{i=1}^k X_i - A\right| \geq dA\right] \leq \frac{1}{\varepsilon d^2} \left(\frac{A}{n^2}\right)^{\frac{2}{3}}.$$

*Proof.* See the Appendix. □

Now we have our main theorem.

**Theorem 3.** *For  $0 < \varepsilon < 1/2$ , if  $BC(v) = n^2/t$ ,  $t \geq 1$ , then with probability more than  $1 - \left(1 + \frac{1}{(2c-1)^2}\right)\varepsilon$ ,  $BC$  can be estimated within a factor of  $1/(\varepsilon t^{1/3})$  with  $\varepsilon t^{2/3}$  samples.*

*Proof.* We first estimate the probability that our algorithm terminates with  $k = \varepsilon(\frac{n^2}{A})^{2/3}$  samples. Lemma 2 guarantees that this probability is less than  $\varepsilon^3/(c - \varepsilon)^2$ . Since  $\varepsilon < 1/2$ , we have

$$\frac{\varepsilon^3}{(c - \varepsilon)^2} \leq \varepsilon \frac{1/4}{(c - 1/2)^2} = \frac{\varepsilon}{(2c - 1)^2}.$$

We next estimate the probability that our algorithm fails to estimate within the factor. By setting  $d = 1/(\varepsilon t^{1/3})$ , Lemma 3 guarantees that the probability of such failure is at most  $\varepsilon$ , if the number of samples  $k \geq \varepsilon(\frac{n^2}{A})^{2/3} = \varepsilon t^{2/3}$ .

Note that except the above two cases, there is no other case when our algorithm will fail to estimate. By subtracting the probabilities, our algorithm will succeed with probability more than  $1 - \varepsilon - \frac{1}{(2c-1)^2}\varepsilon$ . □

By setting  $c = 1$  in Theorem 3, we have the following corollary, which has better parameters compared to Theorem 3 in [4].

**Corollary 2.** *For  $0 < \varepsilon < 1/2$ , if  $BC(v) = n^2/t$ ,  $t \geq 1$ , then with probability more than  $1 - 2\varepsilon$ ,  $BC$  can be estimated within a factor of  $1/(\varepsilon t^{1/3})$  with  $\varepsilon t^{2/3}$  samples.*

## 6 Adaptive Samplings on Pairs of Vertices

In this section we will propose our BC approximation by adaptive samplings on pairs of vertices, and analyze its performance.

## 6.1 The Approximation Algorithm

Riondato et al. [22, 23] give the idea to approximate BC by sampling on shortest paths between a pair of vertices. The number of samplings are calculated beforehand by using Vapnik-Chervonenkis theory. Thus it is *not* adaptive. Riondato's approximation can achieve very small error  $\varepsilon$  with very high successful probability, i.e.,  $1 - \delta$ . However, the number of samplings can be very high, i.e.,  $\propto (\ln 1/\delta)/\varepsilon^2$ . This is because Riondato's algorithm can give simultaneous probabilistic guarantees on the approximation for each vertex, not only one target vertex. Thus it is natural to need more samplings. But it is still interesting to investigate the possibility of adaptive samplings not only on single vertices. Riondato's ABRA algorithm [24] gives the idea to sample on pairs of vertices. Our idea is to merge Bader's [4] and Riondato's [22, 23, 24] methods. We propose Algorithm 1 to estimate the BC value of the vertex  $t$ .

---

**Algorithm 1** Betweenness Centrality Approximation Algorithm Based on Adaptive Sampling on Pairs of Vertices

---

**Require:** Graph  $G = (V, E)$ . Target vertex  $t \in V$ .

**Ensure:** Approximated  $BC(t)$ .

$S \leftarrow 0, k \leftarrow 0$ .

**while**  $S \leq cn$  for some constant  $c \geq 1$  **do**

    Choose a pair of different vertices  $(u, v)$  in the graph.

    Find all the shortest paths from  $u$  to  $v$ . Denote by  $\sigma_{uv}$  the number of the paths.

    Denote by  $\sigma_{uv}(t)$  the number of the above paths that pass through  $t$ .

$S \leftarrow S + \sigma_{uv}(t)/\sigma_{uv}$ .

$k \leftarrow k + 1$ .

**end while**

Return  $n(n-1)S/k$  as the approximated BC of  $t$ .

---

## 6.2 Analysis

Let random variable  $Y_i$  be the value  $\sigma_{uv}(t)/\sigma_{uv}$  of one sample  $(u, v)$ . By a similar manner as  $X_i$  in sampling on vertices, we may find the cdf and pdf of  $Y_i$ .

**Lemma 4.** *The cdf of  $Y_i$  is*

$$\Pr[Y_i \leq x] = \begin{cases} 0 & x \leq 0, \\ 1 - \left(1 - \frac{x}{A}\right)^{n(n-1)-1} & 0 < x < A, \\ 1 & x \geq A. \end{cases}$$

*Therefore, the pdf of  $Y_i$  is*

$$p(x) = \begin{cases} \frac{n(n-1)-1}{A} \left(1 - \frac{x}{A}\right)^{n(n-1)-2} & 0 < x < A, \\ 0 & \text{otherwise.} \end{cases}$$

The mean and variance of  $Y_i$  can also be found analogously.

**Corollary 3.**

$$E[Y_i] = \frac{A}{n(n-1)}, \quad \text{Var}[Y_i] = \frac{n(n-1)-1}{n^2(n-1)^2(n(n-1)+1)} A^2.$$

Like sampling on vertices, we need two lemmas to help prepare the proofs.

**Lemma 5.** *Let  $k = \varepsilon \left(\frac{n^2(n-1)}{A}\right)^{\frac{2}{3}}$ . Then*

$$\Pr\left[\sum_{i=1}^k Y_i \geq cn\right] \leq \frac{\varepsilon^3}{(c-\varepsilon)^2}.$$

*Proof.* See the Appendix. □

**Lemma 6.** Let  $k \geq \varepsilon \left( \frac{n^2(n-1)}{A} \right)^{\frac{2}{3}}$  and  $d > 0$ . Then

$$\Pr \left[ \left| \frac{n(n-1)}{k} \sum_{i=1}^k Y_i - A \right| \geq dA \right] \leq \frac{1}{\varepsilon d^2} \left( \frac{A}{n^2(n-1)} \right)^{\frac{2}{3}}.$$

*Proof.* See the Appendix. □

Now we can summarize the performance result of sampling on pairs of vertices.

**Theorem 4.** For  $0 < \varepsilon < 1/2$ , if  $BC(v) = n^2/t$ ,  $t \geq 1$ , then with probability more than  $1 - \left(1 + \frac{1}{(2c-1)^2}\right) \varepsilon$ ,  $BC$  can be estimated within a factor of  $\frac{1}{\varepsilon} \left( \frac{1}{t(n-1)} \right)^{\frac{1}{3}}$  with  $\varepsilon t^{\frac{2}{3}}(n-1)^{\frac{1}{3}}$  samples.

*Proof.* We first estimate the probability that our algorithm terminates with  $k = \varepsilon \left( \frac{n^2(n-1)}{A} \right)^{2/3}$  samples. Lemma 2 guarantees that this probability is less than  $\varepsilon^3/(c-\varepsilon)^2$ . Since  $\varepsilon < 1/2$ , we have

$$\frac{\varepsilon^3}{(c-\varepsilon)^2} \leq \varepsilon \frac{1/4}{(c-1/2)^2} = \frac{\varepsilon}{(2c-1)^2}.$$

We next estimate the probability that our algorithm fails to estimate within the factor. By setting  $d = \frac{1}{\varepsilon} \left( \frac{1}{t(n-1)} \right)^{\frac{1}{3}}$ , Lemma 3 guarantees that the probability of such failure is at most  $\varepsilon$ , if the number of samples  $k \geq \varepsilon \left( \frac{n^2(n-1)}{A} \right)^{2/3} = \varepsilon t^{2/3}(n-1)^{\frac{1}{3}}$ . □

Similarly, we have the following corollary.

**Corollary 4.** For  $0 < \varepsilon < 1/2$ , if  $BC(v) = n^2/t$ ,  $t \geq 1$ , then with probability more than  $1 - 2\varepsilon$ ,  $BC$  can be estimated within a factor of  $\frac{1}{\varepsilon} \left( \frac{1}{t(n-1)} \right)^{\frac{1}{3}}$  with  $\varepsilon t^{\frac{2}{3}}(n-1)^{\frac{1}{3}}$  samples.

Compared with sampling on vertices, the factor is much less  $\frac{1}{\varepsilon} \left( \frac{1}{t(n-1)} \right)^{\frac{1}{3}} < 1/(\varepsilon t^{1/3})$ , but we use more samples. Fortunately, sampling on pairs has less time cost than sampling on vertices, i.e., it is not necessary to calculate the single source shortest paths (SSSP) to all the vertices in the graph. When the destined vertex is reached, the SSSP calculation can stop earlier without exhausting all the vertices in the graph. However, sampling on vertices requires SSSP to all the vertices for each sample. In this light, we believe sampling on pairs of vertices has better performance than sampling on vertices. We will use simulations on real-world data to verify this fact.

## 7 Evaluations

In this section, we will present our experimental results of revised Bader's method and our proposed algorithms. We will consider the accuracy of the estimation and running time, i.e., the approximation factors and the number of samplings.

*Experiment setup:* We implemented our algorithm, and used the implementation of Brandes' [7] from igraph [10]. All the implementations were written in C, and were tested on uniform system: 4-core Intel Xeon Processors with 16GB RAM, and SMP Linux with kernel version 3.19.8. Each measurement was averaged over 10 instances.

*Datasets:* We use Stanford Large Network Dataset Collection (SLNDC) [18]. We focus on two real-world graphs, `oregon1-010331` and `ego-Facebook`. Note that `oregon1-010331` has been used by Riondato [22], and `ego-Facebook` has fewer vertices but denser edges. Table 1 gives details about the two graphs.

	$ V $	$ E $	avg. clustering coef.
oregon1-010331	10670	22002	0.2970
ego-Facebook	4039	88234	0.6055

Table 1: Details from SLNDC.

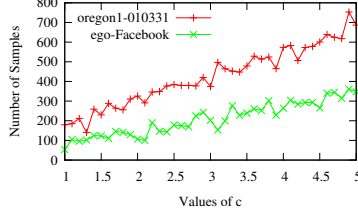


Figure 1:  $c \sim \#$  samples.

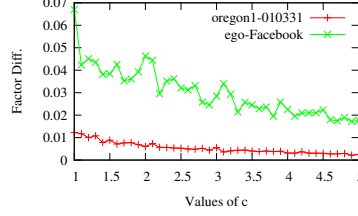


Figure 2:  $c \sim$  factor diff.

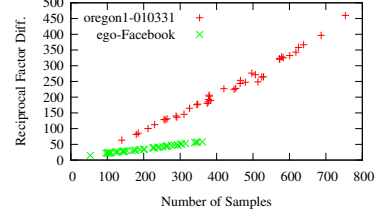


Figure 3:  $\#$  samples  $\sim 1/\text{factor diff.}$

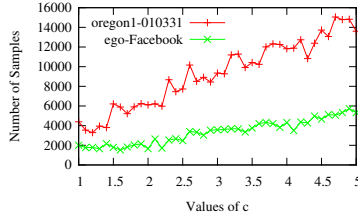


Figure 4:  $c \sim \#$  samples.

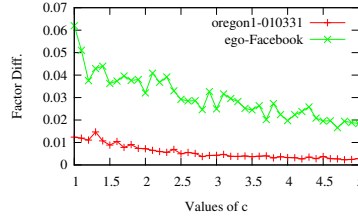


Figure 5:  $c \sim$  factor diff.

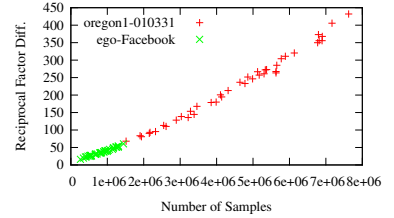


Figure 6:  $\#$  samples  $\sim 1/\text{factor diff.}$

## 7.1 Revised Bader's Algorithm

We present the results for our revised Bader's algorithm. Figure 1 gives that the number of samplings increases if the parameter  $c$  in the algorithm increases. The growth is almost linear, since the curves present steady slopes. This is because in the algorithm, higher  $c$  gives larger threshold when the loop terminates, and hence admits more samplings. Note that the range of  $c$  starts from 1.0. That is different from [4], which required  $c > 2$ . The total number of samplings is no more than 800 for both graphs, which is much less than the number of vertices.

Figure 2 gives that the approximation factor difference decreases if  $c$  increases. The factor different is defined as the distance between the factor and 1. Clearly if the factor is 1, then our estimation gives the exact correct BC value. This is presented in Theorem 3. That is, if the number of samples  $S = \epsilon t^{2/3}$ , then the factor is less than  $b = 1/(\epsilon t^{1/3}) = t^{1/3}/S$ .  $b$  should decrease as  $S$  increases. It also indicates that the revised Bader's algorithm gives better approximations for sparse graphs. The factor differences are all less than 7% for both cases.

Figure 3 gives that the reciprocal of factor difference is almost linearly related to the number of samples. From Theorem 3, the upper bound  $b$  of factor satisfies  $b = t^{1/3}/S$ , or  $1/b \propto S$ . Figure 3 indicates that the actual reciprocal of factor difference seems to be proportional to the number of samplings as well.

Figures 2 and 3 together indicates that the revised Bader's algorithm still works well for  $1 < c < 2$ .

## 7.2 Samplings on Pairs of Vertices

Figure 4 gives that the number of samplings increases if the parameter  $c$  in the algorithm increases. The growth is almost linear, since the curves present steady slopes. Similarly to the first algorithm, this is because in the algorithm, higher  $c$  gives larger threshold when the loop terminates, allowing more samplings. The



number of samplings for graph with more vertices grows faster, since based on Theorem 4, the number of samples is proportional to  $(n - 1)^{1/3}$ , where  $n$  denotes the number of vertices. The sampled paths are no more than 16000 for both graphs.

Figure 5 gives that the approximation factor difference decreases if  $c$  increases. This is presented in Theorem 4. That is, if the number of samples  $S' = \varepsilon t^{2/3}(n - 1)^{1/3}$ , then the factor is less than  $b' = 1/(\varepsilon(t(n - 1))^{1/3}) = t^{1/3}/S'$ .  $b'$  should decrease as  $S'$  increases. It also indicates that our algorithm gives better approximations for sparse graphs. The factor differences are all less than 5% for both cases, which is slightly better than the previous revised Bader's.

Figure 6 gives that the reciprocal of factor difference is almost linearly related to the number of samples. From Theorem 4, the upper bound  $b'$  of factor satisfies  $b' = t^{1/3}/S'$ , or  $1/b' \propto S'$ . It also indicates that the actual reciprocal of factor difference seems to be proportional to the number of samplings as well.

## 8 Future Works

One open problem which is revealed in the experiments is that it seems the reciprocal of factor difference is proportional to the number of samplings. The question is whether this conjecture can be shown by theory. Theorems 3 and 4 both only discuss the upper bound of factors. How to more tightly bound the factors still needs further research.

## 9 Conclusions

In this project, we have revisited Bader's BC estimating algorithm, given more tight bounds on the estimations, and improved the performance of the approximation algorithm, i.e., fewer samples and lower factors. We have also proposed an *adaptive* algorithm with samplings on pairs of vertices. With rigorous reasoning, we have shown that our algorithm can also give bounded approximation factors and number of samplings. To evaluate our results, we have also conducted extensive experiments using real-world datasets. It turns out that both the algorithms can achieve ideal performance. We have verified that our work can improve the current BC approximation algorithm with better parameters, i.e., higher successful probabilities, lower factors and fewer samples.

## 10 Acknowledgement

We are indebted for Matteo Riondato, Vikas Kumar and Rishi Ranjan Singh for their suggestions and comments on this paper.

## References

- [1] ABBOUD, A., GRANDONI, F., AND WILLIAMS, V. V. Subcubic equivalences between graph centrality problems, apsp and diameter. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms* (2015), SIAM, pp. 1681–1697.
- [2] AGARWAL, M., SINGH, R. R., CHAUDHARY, S., AND IYENGAR, S. Bolt: Efficient betweenness ordering in almost linear time. *arXiv preprint arXiv:1409.6470* (2014).
- [3] AGARWAL, M., SINGH, R. R., CHAUDHARY, S., AND IYENGAR, S. An efficient estimation of a nodes betweenness. In *Complex Networks VI*. Springer, 2015, pp. 111–121.
- [4] BADER, D. A., KINTALI, S., MADDURI, K., AND MIHAIL, M. Approximating betweenness centrality. In *Algorithms and Models for the Web-Graph*. Springer, 2007, pp. 124–137.

- [5] BADER, D. A., AND MADDURI, K. A graph-theoretic analysis of the human protein-interaction network using multicore parallel algorithms. *Parallel Computing* 34, 11 (2008), 627–639.
- [6] BARTHELEMY, M. Betweenness centrality in large complex networks. *The European Physical Journal B-Condensed Matter and Complex Systems* 38, 2 (2004), 163–168.
- [7] BRANDES, U. A faster algorithm for betweenness centrality\*. *Journal of Mathematical Sociology* 25, 2 (2001), 163–177.
- [8] BRANDES, U. On variants of shortest-path betweenness centrality and their generic computation. *Social Networks* 30, 2 (2008), 136–145.
- [9] CHEHREGHANI, M. H. An efficient algorithm for approximate betweenness centrality computation. *The Computer Journal* (2014), bxu003.
- [10] CSARDI, G., AND NEPUSZ, T. The igraph software package for complex network research. *InterJournal, Complex Systems* 1695, 5 (2006), 1–9.
- [11] CUI, B., SHAO, X., AND ZHANG, Z. Assessment of flow paths and confluences for saltwater intrusion in a deltaic river network. *Hydrological Processes* (2015).
- [12] FAIRBANKS, J. P., KANNAN, R., PARK, H., AND BADER, D. A. Behavioral clusters in dynamic graphs. *Parallel Computing* (2015).
- [13] FREEMAN, L. C. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.
- [14] GEISBERGER, R., SANDERS, P., AND SCHULTES, D. Better approximation of betweenness centrality. In *ALENEX* (2008), SIAM, pp. 90–100.
- [15] HAGE, P., AND HARARY, F. Eccentricity and centrality in networks. *Social networks* 17, 1 (1995), 57–63.
- [16] KOURTELLIS, N., ALAHAKOON, T., SIMHA, R., IAMNITCHI, A., AND TRIPATHI, R. Identifying high betweenness centrality nodes in large social networks. *Social Network Analysis and Mining* 3, 4 (2013), 899–914.
- [17] LEE, M.-J., CHOI, S., AND CHUNG, C.-W. Efficient algorithms for updating betweenness centrality in fully dynamic graphs. *Information Sciences* 326 (2016), 278–296.
- [18] LESKOVEC, J., AND KREVL, A. Snap datasets: Stanford large network dataset collection. *Stanford University* (2014).
- [19] MADDURI, K., EDIGER, D., JIANG, K., BADER, D., CHAVARRIA-MIRANDA, D., ET AL. A faster parallel algorithm and efficient multithreaded implementations for evaluating betweenness centrality on massive datasets. In *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on* (2009), IEEE, pp. 1–8.
- [20] MAGAIA, N., FRANCISCO, A. P., PEREIRA, P., AND CORREIA, M. Betweenness centrality in delay tolerant networks: A survey. *Ad Hoc Networks* (2015).
- [21] PUZIS, R., ZILBERMAN, P., ELOVICI, Y., DOLEV, S., AND BRANDES, U. Heuristics for speeding up betweenness centrality computation. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)* (2012), IEEE, pp. 302–311.
- [22] RIONDATO, M., AND KORNAROPOULOS, E. M. Fast approximation of betweenness centrality through sampling. In *Proceedings of the 7th ACM international conference on Web search and data mining* (2014), ACM, pp. 413–422.

- [23] RIONDATO, M., AND KORNAROPOULOS, E. M. Fast approximation of betweenness centrality through sampling. *Data Mining and Knowledge Discovery* 30, 2 (2016), 438–475.
- [24] RIONDATO, M., AND UPFAL, E. Abra: Approximating betweenness centrality in static and dynamic graphs with rademacher averages. *KDD'16, available at arXiv preprint arXiv:1602.05866* (2016).
- [25] SABIDUSSI, G. The centrality index of a graph. *Psychometrika* 31, 4 (1966), 581–603.
- [26] SHIMBEL, A. Structural parameters of communication networks. *The bulletin of mathematical biophysics* 15, 4 (1953), 501–507.
- [27] SIVARANJANI, S., AND SIVAKUMARI, S. Mitigating serial hot spots on crime data using interpolation method and graph measures. *International Journal of Computer Applications* 126, 7 (2015).

## 11 Appendix

### 11.1 Proof of Lemma 1

*Proof.* Note that  $\sum_{v_i \in V, v_i \neq v} X_i = A$ . Suppose the interval  $(0, A)$  is randomly divided into  $n$  sub-intervals, each of which represents the length of  $X_i$ . If the  $i$ -th sub-interval is larger than  $x$ , then all the  $n - 1$  cutting points are distributed outside a sub-interval, whose length is  $x$ . Formally, we have

$$\Pr[X_i > x] = \left(1 - \frac{x}{A}\right)^{n-1}$$

and  $\Pr[X_i \leq x] = 1 - \Pr[X_i > x]$  gives the desired cdf. By computing the derivative over  $0 < x < A$ , we have the pdf as desired.  $\square$

### 11.2 Proof of Corollary 1

*Proof.*

$$\begin{aligned} E[X_i] &= \int_0^A x \frac{n-1}{A} \left(1 - \frac{x}{A}\right)^{n-2} dx \\ &= -x \left(1 - \frac{x}{A}\right)^{n-1} \Big|_0^A + \int_0^A \left(1 - \frac{x}{A}\right)^{n-1} dx \\ &= -\frac{A}{n} \left(1 - \frac{x}{A}\right)^n \Big|_0^A = \frac{A}{n}. \\ E[X_i^2] &= \int_0^A x^2 \frac{n-1}{A} \left(1 - \frac{x}{A}\right)^{n-2} dx \\ &= -x^2 \left(1 - \frac{x}{A}\right)^{n-1} \Big|_0^A + 2 \int_0^A x \left(1 - \frac{x}{A}\right)^{n-1} dx \\ &= \frac{2A}{n} \int_0^A \left(1 - \frac{x}{A}\right)^n dx = \frac{2A^2}{n(n+1)}. \\ \text{Var}[X_i] &= E[X_i^2] - E[X_i]^2 = \frac{2A^2}{n(n+1)} - \frac{A^2}{n^2} = \frac{n-1}{n^2(n+1)} A^2. \end{aligned}$$

$\square$

Since  $0 \leq A \leq n^2$ , it is clear our model agrees with [4]’s bounds on  $E[X_i]$  and  $\text{Var}[X_i]$ . However for  $E[X_i^2]$ , our model gives the upper bound:  $E[X_i^2] \leq 2A$ . This can be explained: it is possible for  $\text{Var}[X_i]$  to reach  $A$  very closely (e.g., when  $A$  is very close to  $n^2$ ), and since  $E[X_i^2] = \text{Var}[X_i] + E[X_i]^2 = \text{Var}[X_i] + A^2/n^2$ , it is possible that  $\text{Var}[X_i]$  exceeds  $A$ . Luckily neither this paper nor Bader’s work uses this bound on  $E[X_i^2]$  in the rest reasoning.

### 11.3 Proof of Lemma 2

*Proof.* Note that  $BC(v) = A \leq n^2$ . Then we have

$$\begin{aligned}
& \Pr\left[\sum_{i=1}^k X_i \geq cn\right] = \Pr\left[\sum_{i=1}^k \left(X_i - \frac{A}{n}\right) \geq cn - \frac{kA}{n}\right] \\
& = \Pr\left[\sum_{i=1}^k \left(X_i - \frac{A}{n}\right) \geq cn - \varepsilon n \left(\frac{A}{n^2}\right)^{\frac{1}{3}}\right] \\
& \leq \Pr\left[\sum_{i=1}^k \left(X_i - \frac{A}{n}\right) \geq cn - \varepsilon n\right] \quad (A \leq n^2) \\
& \leq \Pr\left[\bigvee_{i=1}^k \left(X_i - \frac{A}{n}\right) \geq \frac{c-\varepsilon}{k}n\right] \quad (\text{at least one } X_i \text{ exceeds the average}) \\
& \leq \sum_{i=1}^k \Pr\left[\left(X_i - \frac{A}{n}\right) \geq \frac{c-\varepsilon}{k}n\right] \quad (\text{union bound}) \\
& \leq \sum_{i=1}^k \frac{k^2}{(c-\varepsilon)^2 n^2} \text{Var}[X_i] \quad (\text{Chebychev's inequality}) \\
& = \frac{k^2}{(c-\varepsilon)^2 n^2} \sum_{i=1}^k \frac{n-1}{n^2(n+1)} A^2 \quad (\text{Corollary 1}) \\
& = \frac{\varepsilon^3(n-1)}{(c-\varepsilon)^2(n+1)} \leq \frac{\varepsilon^3}{(c-\varepsilon)^2}.
\end{aligned}$$

□

### 11.4 Proof of Lemma 3

*Proof.*

$$\begin{aligned}
& \Pr\left[\left|\frac{n}{k} \sum_{i=1}^k X_i - A\right| \geq dA\right] = \Pr\left[\left|\sum_{i=1}^k X_i - \frac{kA}{n}\right| \geq \frac{k d A}{n}\right] \\
& = \Pr\left[\left|\sum_{i=1}^k \left(X_i - \frac{A}{n}\right)\right| \geq \frac{k d A}{n}\right] \leq \frac{n^2}{k^2 d^2 A^2} k \text{Var}[X_i] \quad (\text{Chebychev's inequality}) \\
& = \frac{n-1}{(n+1)k d^2} \leq \frac{1}{k d^2} = \frac{1}{\varepsilon d^2} \left(\frac{A}{n^2}\right)^{\frac{2}{3}}.
\end{aligned}$$

□

## 11.5 Proof of Lemma 5

*Proof.*

$$\begin{aligned}
& \Pr\left[\sum_{i=1}^k Y_i \geq cn\right] \\
&= \Pr\left[\sum_{i=1}^k \left(Y_i - \frac{A}{n(n-1)}\right) \geq cn - \frac{kA}{n(n-1)}\right] \\
&= \Pr\left[\sum_{i=1}^k \left(Y_i - \frac{A}{n(n-1)}\right) \geq cn - \varepsilon n \left(\frac{A}{n^2(n-1)}\right)^{\frac{1}{3}}\right] \\
&\leq \Pr\left[\sum_{i=1}^k \left(Y_i - \frac{A}{n(n-1)}\right) \geq cn - \varepsilon n\right] \quad (A \leq n^2) \\
&\leq \Pr\left[\bigvee_{i=1}^k \left(Y_i - \frac{A}{n(n-1)}\right) \geq \frac{c-\varepsilon}{k}n\right] \quad (\text{at least one } Y_i \text{ exceeds the average}) \\
&\leq \sum_{i=1}^k \Pr\left[\left(Y_i - \frac{A}{n(n-1)}\right) \geq \frac{c-\varepsilon}{k}n\right] \quad (\text{union bound}) \\
&\leq \sum_{i=1}^k \frac{k^2}{(c-\varepsilon)^2 n^2} \text{Var}[Y_i] \quad (\text{Chebychev's inequality}) \\
&= \frac{k^2}{(c-\varepsilon)^2 n^2} \sum_{i=1}^k \frac{n(n-1)-1}{n^2(n-1)^2(n(n-1)+1)} A^2 \quad (\text{Corollary 3}) \\
&= \frac{\varepsilon^3(n(n-1)-1)}{(c-\varepsilon)^2(n(n-1)+1)} \leq \frac{\varepsilon^3}{(c-\varepsilon)^2}. \quad (\text{Jensen's inequality})
\end{aligned}$$

□

Alternatively one can use the bound on variance from [4]:  $\text{Var}[Y_i] < A$ . Analogous computation can show the upper bound is still the same.

## 11.6 Proof of Lemma 6

*Proof.*

$$\begin{aligned}
& \Pr\left[\left|\frac{n(n-1)}{k} \sum_{i=1}^k Y_i - A\right| \geq dA\right] \\
&= \Pr\left[\left|\sum_{i=1}^k Y_i - \frac{kA}{n(n-1)}\right| \geq \frac{kdA}{n(n-1)}\right] \\
&= \Pr\left[\left|\sum_{i=1}^k \left(Y_i - \frac{A}{n(n-1)}\right)\right| \geq \frac{kdA}{n(n-1)}\right] \leq \frac{n^2(n-1)^2}{k^2 d^2 A^2} k \text{Var}[Y_i] \quad (\text{Chebychev's inequality}) \\
&= \frac{n(n-1)-1}{(n(n-1)+1)kd^2} \leq \frac{1}{kd^2} = \frac{1}{\varepsilon d^2} \left(\frac{A}{n^2(n-1)}\right)^{\frac{2}{3}}. \quad (\text{Jensen's inequality})
\end{aligned}$$

□